A-PPL: An Accountability Policy Language

Monir Azraoui[1], Kaoutar Elkhiyaoui[1], Melek Önen[1], Karin Bernsmed[2], Anderson Santana De Oliveira[3], and Jakub Sendor[3]

[1] EURECOM, Biot Sophia Antipolis, France
{monir.azraoui, kaoutar.elkhiyaoui, melek.onen}@eurecom.fr
[2] SINTEF ICT, Trondheim, Norway
karin.bernsmed@sintef.no
[3] SAP Labs France, Mougins Sophia Antipolis, France
{anderson.santana.de.oliveira, jakub.sendor}@sap.com

Abstract. Cloud Computing raises various security and privacy challenges due to the customers' inherent lack of control over their outsourced data. One approach to encourage customers to take advantage of the cloud is the design of new accountability solutions which improve the degree of transparency with respect to data processing. In this paper, we focus on accountability policies and propose A-PPL, an accountability policy language that represents machine-readable accountability policies. A-PPL extends the PPL language by allowing customers to define additional rules on data retention, data location, logging and notification. The use of A-PPL is illustrated with a use case where medical sensors collect personal data which are then stored and processed in the cloud. We define accountability obligations related to this use case and translate them into A-PPL policies as a proof of concept of our proposal.

1 Introduction

Cloud computing marks a shift in the way organizations and individuals consume technology. Customers outsource large amount of data into the cloud and delegate the implementation of numerous security and privacy controls over these data to the cloud service provider. This new paradigm hence raises accountability concerns; in particular, business customers perceive data lock-in, loss of governance and non-compliance as major risks associated with the cloud [11].

As defined in [14], accountability concerns data stewardship regimes in which organizations that are entrusted with personal and business confidential data are responsible and liable for processing, sharing, storing and otherwise using the data according to contractual and legal requirements from the time the data is collected until when it is destroyed (including onward transfer to and from third-parties). In such a setting, clarifying the accountability relationships, i.e. who is responsible to whom and for what, with clear organizational policies will help overcome barriers to data governance in the cloud. Appropriate policies mitigate risks, provided that reliable tools to enforce them and to monitor their effectiveness are in place to allow audits.

In this paper, we are interested in machine-readable representations of policies expressing accountability obligations. Such policies will help service providers deploy their automatic enforcement when personal data is processed. We propose a new policy language, A-PPL, that enables the expression of the accountability obligations. A-PPL (shorthand for Accountable PPL) extends the existing PPL language [15] by allowing the expression of accountability obligations such as rules on data retention, data location, logging and notification. We first present a number of design requirements for an accountability policy language. We then describe PPL and its limitations. A-PPL which aims at addressing these shortcomings is further introduced and finally illustrated with a healthcare use case.

2 Policy Language Requirements

Following an analysis of accountability relationships between cloud actors in [1, 5], we derive a collection of eight requirements for the design of an accountability policy language. The new policy language should express data handling rules that correspond to (R1) Privacy Policies, (R2) Access Control Rules, (R3) Usage Control Rules and (R4) Data Retention Periods. The requirements related to accountability needs, often absent in existing policy languages, concern (R5) Reporting and Notification, (R6) Data Location Rules, (R7) Auditability and (R8) Logging. One may argue that these requirements can be expressed and enforced using multiple languages at different levels of the cloud technology stack. We advocate that centralizing these concerns in a single policy will improve the accountability of the actors processing personal data in the cloud, while decreasing the loss of governance, as policies will not be diluted across the service provisioning chain. Besides, rather than imposing a new language, we aim at choosing an existing language that fulfills the above requirements the best and which is extensible enough to add accountability extensions to it.

3 A-PPL: An Accountable Policy Language

In [1, 5], we conduct an analysis of existing policy languages against the design requirements we identified in Section 2. As a result, we choose the PrimeLife Policy Language (PPL) as the best candidate language since it covers most of the requirements and PPL is an extensible language. Therefore, we create new extensions to PPL to build A-PPL.

3.1 PrimeLife Policy Language (PPL) and its limitations

PPL [15] was proposed by the European ICT PrimeLife[4] project to specify machine-readable privacy policies. PPL extends XACML [12] by defining a new obligation and authorization syntax. In PPL, an obligation is expressed using the pair Trigger-Action. Triggers are events related to an obligation and filtered

[4] http://www.primelife.eu/

by conditions. Triggers fire Actions that are performed by the data controller. The complete list of available PPL Triggers and Actions can be found in [15]. Authorizations define the actions that the data controller is allowed or prohibited to perform: (i) authorization for purposes, allows the data controller to perform actions for a particular set of well-stated usage purposes; (ii) authorization for downstream usage, allows the forwarding of collected data to third-parties.

While PPL meets most of the requirements we identify in Section 2 (such as privacy policies (R1), access and usage control rules (R2)), it falls short in meeting other requirements such as data location (R6) or auditability (R7). Having identified the limitations of PPL, we propose in Section 3.2 our accountable policy language A-PPL that extends PPL based on each requirement.

3.2 The syntax of A-PPL

In this section we present the extensions we add to PPL to create A-PPL. Note that we maintain the overall structure of PPL. More details can be found in [5].

Roles. A-PPL identifies different actors that can hold different roles: data subject, data controller, data processor and data protection authority. PPL already defines the first three roles. We create the auditor role. To make the identification of roles explicit in A-PPL, we introduce a role attribute identifier `subject:role` to be included as an attribute of the standard XACML element <`Subject`>.

Access Control Rules (R2). We introduce two new triggers which condition the execution of an obligation based on the result of an access decision. More specifically, we propose `TriggerPersonalDataAccessPermitted` and `Trigger-PersonalDataAccessDenied` that occur when the evaluation of the access control on the targeted data results in "Permit" and "Deny", respectively.

Data Retention (R4). PPL provides an element `Purpose` that allows to specify for which purpose a piece of data can be collected or accessed. In A-PPL, we define the `duration` attribute for `Purpose` that allows to specify for how long the data can be processed for a particular purpose. This attribute implies that when all durations for each purpose have expired, the data has to be deleted, since the data cannot be used for any purpose anymore.

Reporting and Notification (R5). We modify the existing PPL `Action-NotifyDataSubject` element and call the newly created action `ActionNotify`. We provide the attribute `recipient` that allows to indicate the recipient of the notification and the attribute `type` that specifies the type of notification to be sent to the recipient.

Controlling Data Location (R6). We propose in A-PPL a standard identifier `region` to specify the location in the `AuthzUseForPurpose` element. Thus we define the authorized locations of processing of data for particular purposes.

Auditability (R7). We propose two extensions that relate to audits and collection of evidence. This evidence collection is governed by a new A-PPL trigger `TriggerOnEvidenceRequestReceived`, and a new A-PPL action `Action-EvidenceCollection`. The combination of these two elements initiates the evidence collection by the data controller.

Logging (R8). We extend the `ActionLog` element in A-PPL. We introduce several parameters to make explicit which information about an event needs to be logged: a timestamp, the action that is performed on the data, the identity of the subject who performed the action and the purpose of the action (e.g. `marketing`). Other details can also be written in the logs such as some security flags that may state whether the log entry is encrypted.

Note that possible conflicts between accountable obligations are not addressed by A-PPL. As studied in our report [5], we consider that these conflicts are solved before mapping the obligations to A-PPL policies.

3.3 A-PPLE (A-PPL Engine): the extension of the PPL Engine

The policy engine supporting PPL was originally designed in PrimeLife project [15]. We briefly present the new architecture of A-PPLE whereby we extend and modify the PPL engine's architecture to implement the new features of the A-PPL policy language. The core elements of A-PPLE are the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP). While the PEP acts as an orchestrator of the enforcement process and interfaces with the Web Services, the PDP is the component where the access control decision is taken. PDP relies on the access control engine implementation based on HERAS [9] for the evaluation of the XACML part of an A-PPL policy. Apart from the standard attribute-based access control, the other information evaluated by the PDP at the step of access control decision is usage authorization. The PEP coordinates two modules: the Event and Obligation Handlers. The functionality of the Event Handler is to fire the events related to the personal data lifecycle, e.g. when data is deleted from the Personal Data store or when it is shared with the third parties. The Obligation Handler keeps track of the triggers that are part of the obligation statements in the A-PPL policy. Once the events are observed, the action associated with the obligation is activated by the Obligation Engine. We also define an additional and central component for handling the audit requests which will facilitate the process of retrieving the necessary information from the systems (logs related to obligations, notifications, access control decisions and personal data lifecycle). Finally, each component in the engine architecture (Obligation Handler, Event Handler, PDP and PEP) are linked with the logging Handler to record all data sensitive actions in a non-repudiable manner. More details on the new A-PPL engine can be found in [1].

4 Example of Use of A-PPL

In this section we present a use case that illustrates how accountability obligations can be expressed using A-PPL. The use case that we describe is a healthcare system that will be used to support elderly people by analyzing medical data collected by sensors. We investigate a case where medical data from the sensors will be exchanged between the elderly, their families and friends, hospital caregivers and healthcare personnel. The proposed solution is the M Platform, which is a cloud-based service for medical sensor data collection, processing, storage and visualization. The sensor data will be transmitted to the cloud where they will be further processed and stored. The M Platform is offered to the hospital as a service from a European software and service provider M, which has outsourced both the initial storage of data collected through the sensors placed by hospital staff (Cloud x, which is provided by X) as well as the long-term data storage and back-up procedures (Cloud y, which is provided by Y). Further details about this healthcare system can be found in [1, 3, 2]. To comply with the European Data Protection Directive [7], as well as with the contractual relationships that must exist between the involved actors, a number of accountability obligations can be derived for the healthcare use case. Here we outline some of the most prominent ones with their mapping into A-PPL statements. Further details on the obligations can be found in [3].

Obligation 1: The right to access, correct and delete personal data. The hospital must ensure that the patients have read, write and delete access to their personal data that have been collected and stored in the cloud. The right to delete is expressed in Listing 1.1 as XACML rules that A-PPL is built upon. Similar rules can be specified for read and write access.

Listing 1.1: Access control

```
<Rule Effect="Permit" RuleId="DS_access">
 <Target>
  <Subject>
   <SubjectMatch MatchId="function:string-equal">
    <AttributeValue DataType="string">Data Subject</AttributeValue>
    <SubjectAttributeDesignator DataType="string"
       AttributeId="subject:role-id"/>
   </SubjectMatch>
  </Subject>
  <Action>
   <ActionMatch MatchId="string-equal">
    <AttributeValue DataType="string">delete</AttributeValue>
    <ActionAttributeDesignator DataType="string"
      AttributeId="action-id"/>
   </ActionMatch>
  </Action>
 </Target>
</Rule>
```

Obligation 2: Purpose of processing. The hospital must make sure that the patients' personal data is only processed for specific, explicit and legitimate purposes. Listing 1.2 shows an A-PPL statement that specifies the purposes and their respective durations of use of the collected data.

Listing 1.2: Authorization for the specified list of purposes

```
<a-ppl:AuthzUseForPurpose>
  <a-ppl:Purpose duration=2Y region=Europe>diagnosis</a-ppl:Purpose>
  <a-ppl:Purpose duration=5Y region=Europe>research</a-ppl:Purpose>
</a-ppl:AuthzUseForPurpose>
```

Obligation 3: Breach notification. In case of security or personal data breaches, cloud providers X and Y must notify M, which in turn must notify the hospital and the hospital must notify the patients. Listing 1.3 shows an example of the use of the `ActionNotify` element whereby the data controller is hold responsible for notification in case of a policy violation[5] or a loss of data.

Listing 1.3: Notify the data subject in case of a breach

```
<Obligation>
 <TriggersSet>
   <TriggerOnPolicyViolation/>
   <TriggerOnDataLost/>
 </TriggersSet>
 <ActionNotify>
   <Media>e-mail</Media>
   <Address>data-subject@example.com</Address>
   <Recipients>Patient:Data subject</Recipients>
   <Type>Policy Violation</Type>
 </ActionNotify>
</Obligation>
```

Obligation 4: Evidence of the correct and timely deletion of personal data. Cloud providers X and Y must be able to provide evidence to the platform provider M, and M must be able to provide evidence to the hospital on the correct and timely deletion of personal data. Listing 1.4 shows an example of logging of the delete action and Listing 1.5 collects these logs as evidence.

Listing 1.4: Log deletion

```
<Obligation>
 <TriggerPersonal-
DataDeleted>
 </TriggerPersonal-
DataDeleted>
 <ActionLog>
  <Timestamp/>
  <Action/>
  <Purpose/>
  <Subject/>
  <Resource/>>
 </ActionLog>
</Obligation>
```

Listing 1.5: Evidence of data deletion

```
<Obligation>
 <TriggerEvidenceRequestReceived>
 </TriggerEvidenceRequestReceived>
 <ActionEvidenceCollection>
  <Evidence>
   <Attribute AttributeId="evidence-type">
    <AttributeValue>Logs of deletion</AttributeValue>
   </Attribute>
  </Evidence>
  <Resource>
   <Attribute AttributeId="resource-id">
    <AttributeValue>Personal Data</AttributeValue>
   </Attribute>
  </Resource>
 </ActionEvidenceCollection>
</Obligation>
```

Obligation 5: Location of processing. Cloud providers X and Y, as well as the M Platform provider have contractual obligations towards their respective cus-

[5] Violations are detected by an external tool that takes A-PPL policies as inputs.

tomers on the location of the data processing. Listing 1.2 presents an example of how to specify authorized location of processing.

5 Related Work

Various work design a language for specifying obligations such as [12, 15, 10, 6, 13]. But little focus has been put on a language for accountability. For instance, [10] proposes an extension of the XACML architecture to enable the enforcement of obligations related to access control decisions.

Contemporaneous work by Butin et al. [4] leverages PPL to design logs for accountability. They identify the lack of expressiveness of PPL `ActionLog` which does not provide sufficient information in the logs. Besides, they discuss the fact that the PPL element `ActionNotifyDataSubject` does not allow to specify the content of the notification. Our accountability language A-PPL proposes a solution for these two above problems.

Similarly, Henze et al. [8] identify location and duration of storage as the two main challenges in cloud data handling scenarios. They use PPL to specify *data annotations* that contain the data handling obligations (e.g "delete after 30 days"). Without giving more details, they propose to extend PPL with elements that specify a maximum and a minimum duration of storage and with an element that restricts the location of data. A-PPL addresses these two challenges and we give in Section 3.2 the details of the extensions that solve these issues.

6 Conclusion

In this paper, we defined the design requirements for an accountability policy language. We presented A-PPL, an extension of PPL, that handles accountability specific requirements such as notification, logging and evidence collection. We also described an architecture of A-PPLE, the policy engine that enforces A-PPL policies. Finally, we extracted several obligations from a healthcare use case and defined the corresponding A-PPL rules.

Our future work will consist in the finalization of A-PPLE and its integration in a real setting that combines enforcement tools (such as an audit system).

7 Acknowledgments

This work was supported by the European Commission's Seventh framework A4Cloud[6] project. We thank Dimitra Stefanatou for her help on the analysis of accountability obligations.

[6] http://www.a4cloud.eu/

References

1. M. Azraoui, K. Elkhiyaoui, M. Önen, K. Bernsmed, A. Santana de Oliveira, and J. Sendor. A-PPL: An Accountability Policy Language. Technical report, 2014.
2. K. Bernsmed, M. Felici, A. S. D. Oliveira, J. Sendor, N. B. Moe, T. Rübsamen, V. Tountopoulos, and B. Hasnain. Use case descriptions. Deliverable, Cloud Accountability (A4Cloud) Project, 2013.
3. K. Bernsmed, H. Kuan, and C. Millard. Deploying Medical Sensor Networks in the Cloud - Accountability Obligations from a European Perspective. *Submitted for publication*, 2014.
4. D. Butin, M. Chicote, and D. Le Métayer. Log design for accountability. In *Security and Privacy Workshops (SPW), 2013 IEEE*, pages 1–7. IEEE, 2013.
5. R.-A. Cherrueau, R. Douence, H. Grall, J.-C. Royer, M. Sellami, M. Südholt, M. Azraoui, K. Elkhiyaoui, R. Molva, M. Önen, A. Garaga, A. S. de Oliveira, J. Sendor, and K. Bernsmed. Policy representation framework. Deliverable (to be published), Cloud Accountability (A4Cloud) Project, 2013.
6. F. Cuppens and N. Cuppens-Boulahia. Modeling contextual security policies. *International Journal of Information Security*, 7(4):285–305, 2008.
7. European Parliament. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data., 1995.
8. M. Henze, M. Großfengels, M. Koprowski, and K. Wehrle. Towards data handling requirements-aware cloud computing. In *2013 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2013.
9. HERAS AF team. HERAS AF (Holistic Enterprise-Ready Application Security Architecture Framework). http://herasaf.org/.
10. N. Li, H. Chen, and E. Bertino. On practical specification and enforcement of obligations. In *Proceedings of the second ACM conference on Data and Application Security and Privacy*, pages 71–82. ACM, 2012.
11. A. Lin and N.-C. Chen. Cloud computing as an innovation: Percepetion, attitude, and adoption. *International Journal of Information Management*, 32(6):533 – 540, 2012.
12. OASIS Standard. eXtensible Access Control Markup Language (XACML) Version 3.0. 22 January 2013. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, 2013.
13. E. I. Papagiannakopoulou, M. N. Koukovini, G. V. Lioudakis, N. Dellas, J. Garcia-Alfaro, D. I. Kaklamani, I. S. Venieris, N. Cuppens-Boulahia, and F. Cuppens. Leveraging ontologies upon a holistic privacy-aware access control model. In *Foundations and Practice of Security*, pages 209–226. Springer, 2014.
14. S. Pearson, V. Tountopoulos, D. Catteddu, M. Sudholt, R. Molva, C. Reich, S. Fischer-Hubner, C. Millard, V. Lotz, M. Jaatun, R. Leenes, C. Rong, and J. Lopez. Accountability for cloud and other future internet services. In *2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 629–632, 2012.
15. S. Trabelsi, G. Neven, D. Raggett, C. Ardagna, C. Bournez, L. Bussard, M. Bezzi, J. Camenisch, S. de Capitani di VIMERCATI, F. Gey, A. Kuczerawy, S. Meissner, G. Neven, A. Njeh, S. Paraboschi, E. Pedrini, S. Foresti, U. Pinsdorf, F.-S. Preiss, J. Sendor, C. Tziviskou, D. Raggett, T. Roessler, P. Samarati, J. Schallaboeck, S. Short, D. Sommer, M. Verdicchio, and R. Wenning. D5.3.4 - report on design and implementation of the primelife policy language and engine. Deliverable, Primelife Project, 2011.